

DOCUMENT RESUME

ED 368 325

IR 016 551

AUTHOR Salter, Mary
TITLE Exploring Recursion with "LogoWriter" and Teacher's Manual.
INSTITUTION International Society for Technology in Education, Eugene, OR.
REPORT NO ISBN-1-56484-031-X; ISBN-1-56484-032-8
PUB DATE 93
NOTE 102p.
AVAILABLE FROM International Society for Technology in Education, 1787 Agate St., Eugene, OR 97403-1923 (The "Exploring Recursion with LogoWriter" package includes this book, a data disk, and a single-user site license).
PUB TYPE Guides - Classroom Use - Instructional Materials (For Learner) (051) -- Computer Programs (101)
EDRS PRICE MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS *Computer Assisted Instruction; Computer Software; Higher Education; Individualized Instruction; Learner Controlled Instruction; *Learning Activities; *Programming; Resource Materials; Secondary Education; Teaching Methods; Units of Study
IDENTIFIERS *LOGO Programing Language; LogoWriter; *Recursive Programing; Turtle Graphics

ABSTRACT

This document combines a students' manual and a manual for teachers. The students' manual is a self-teaching guided discovery manual for junior high, high school, and college students that provides self-paced explorations into recursion. It includes a disk with lessons that are to be used with the written material. There are eight lessons presented: (1) lesson 1 involves performing experiments and examining Logo procedures that introduce recursion; (2) the objective of lesson 2 is to develop a conceptual model for embedded procedure calls that are nonrecursive; (3) lesson 3 explores simple procedures and creating and diagraming recursive procedures; (4) lesson 4 investigates procedures that involve recursion and procedure inputs and reinforces the concept of helpers in recursion; (5) lesson 5 involves learning to use conditionals to stop recursive procedures and practicing diagramming procedures that use stop rules; and (6) lesson 6 explores and predicts the results of procedures that use embedded recursion and looks at patterns and processes within procedures; (7) the objective of lesson 7 is to explore Logo variables; and (8) lesson 8 provides an introduction to fractals. Each lesson section includes objectives, procedures, and discussion. Data sheets are provided to help students evaluate their knowledge of the lesson content. A chart of relevant LogoWriter keys is appended. The purpose of the teacher's manual is to help teachers support students in their learning of recursion. The manual is divided into four parts: (1) part 1 contains the instructions for using the manual and contains guidelines for assessing student progress; (2) part 2 lists references to reading that will give teachers insight into recursion and the teaching of recursion, as well as activities that can be used with a small group or class; (3) part 3 contains masters of the "data sheets" that correspond to those in the student book; and (4) part 4 includes procedure trees and listings of all the procedures contained on the disk accompanying the student package. A chart of the relevant LogoWriter Keys and a list of 17 suggested readings are appended. (TMK)

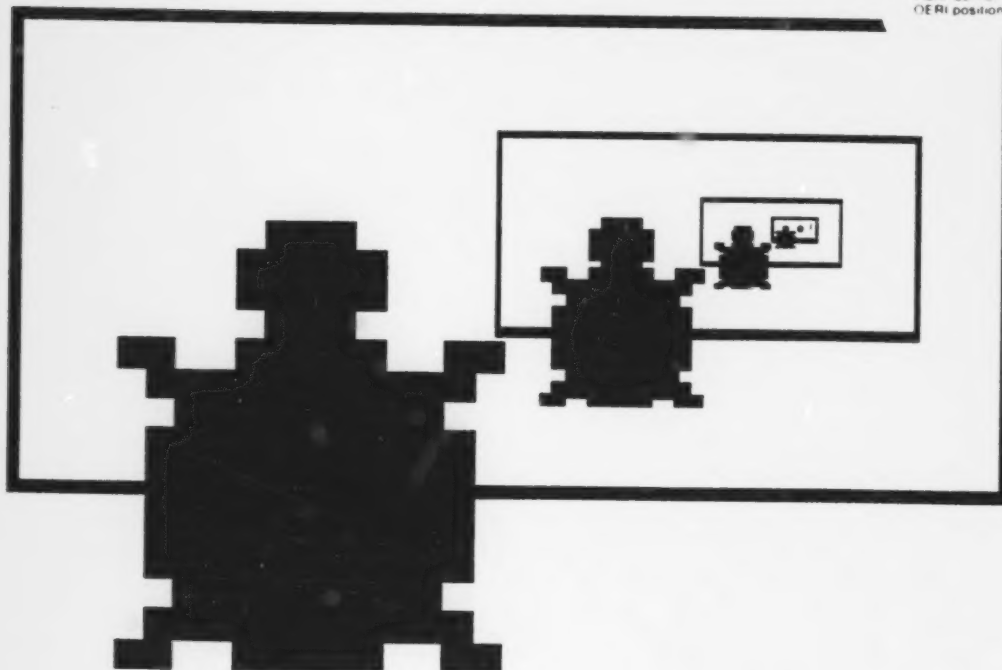
Exploring Recursion

With *LogoWriter*

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)

- ☐ This document has been reproduced as received from the person or organization originating it.
- ☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.



Mary Salter

BEST COPY AVAILABLE

PERMISSION TO REPRODUCE THIS MATERIAL
IN OTHER THAN PAPER COPY HAS BEEN
GRANTED BY D. Moursund.

About the Author

Mary Salter received her M.S. degree in math from Loyola University in New Orleans. She has taught mathematics and computer science at high school and college levels for 15 years and is currently an associate professor of mathematics and computer science at Franciscan University of Steubenville in Steubenville, Ohio. She teaches computer programming and applications courses for preservice teachers and graduate education computer courses for teachers and administrators. She has been a contributor to *Logo Exchange* and has coauthored a text on linear algebra experiments using the DERIVE program.

Technical Editor: Sharon Yoder

Editing and Production Assistance: The Electronic Page

Book Cover: Sharon Yoder

Book Design: Jean Hall

© International Society for Technology in Education, 1993

1787 Agate Street, Eugene, Oregon 97403-1923

Order Desk: 800/336-5191

Phone: 503/346-4414

Fax: 503/346-5890

Compuserve: 70014,2117

Internet: ISTE@Oregon.uoregon.edu

ISBN 1-56484-031-X

Exploring Recursion

With *LogoWriter*

Mary Salter

© International Society for Technology in Education, 1993

From the Publisher

The purchaser of the *Exploring Recursion With LogoWriter* package receives this book, a data disk, and a single-user site license. The *LogoWriter* application is *not* included and is necessary for using the disk files.

The single-user site license gives the one person who is designated as the "primary user" of the *Exploring Recursion With LogoWriter* package the right to make copies of Data Sheets, disk files, and other material necessary for use in the classes he or she teaches. The right to copy materials from the book or the disk does not transfer to participants in classes taught by the primary user. If the single-user site license is purchased by a school or school district, the intent is that one person be designated as the "primary user." If several people are to use the book for teaching, separate packages of *Exploring Recursion With LogoWriter* must be purchased for each individual teacher.

Table of Contents

Preface i

Lesson 1. Preliminaries 1

Lesson 2. Embedded Procedures 9

Lesson 3. Simple Recursion 13

Lesson 4. Recursion Involving Procedure Inputs 19

Lesson 5. Stop Rules 25

Lesson 6. Embedded Recursion 31

Lesson 7. Logo Variables 41

Lesson 8. Introduction to Fractals 47

Appendix A. LogoWriter Keys 53

Preface

A Note to the Student

This book is designed to be a self-teaching guided discovery manual that provides explorations into recursion. It includes a disk with lessons that are to be used along with the material in the book. You should be sitting at the computer as you work through the lessons.

Because the Logo learning environment is often self-directed and self-paced, this book is designed for individual use. You can, however, use it while working together with a small group of other students. If you do work together, it is important that each of you complete the Data Sheet included at the end of each lesson. Extra Data Sheets are available from your teacher.

Questions on the Data Sheets are numbered to correspond to the numbered steps within the lesson. When directed in the reading, you should complete the observations and exercises on the Data Sheet before continuing to the next activity. In the procedure section of each chapter, the numbered items that require you to complete a corresponding activity on the Data Sheet are marked with a $\#$ symbol. Turn in your Data Sheet to your teacher before beginning the next lesson.

The material in this book was designed for use on Apple II, Macintosh, or MS-DOS computers. *LogoWriter* also is required. A color monitor is highly recommended because color is used throughout the lessons.

For convenience, the procedure CS is included in each of the lessons on the accompanying disk. It may be helpful to think of it as ClearScreen, because it is a quick way to clear the screen of text and graphics and return the turtle to the home position.

You may encounter some unfamiliar terms as you read through the lessons. You might want to ask your teacher about some of them immediately or you might try to come to understand them as you work through the lesson. If there are any

terms you still don't understand when you have finished the lesson, be sure to discuss them with your teacher.

You should not be in a hurry to complete the lessons. Repeat activities until you understand the basic concepts. Feel free to explore your own ideas. If you are working with others, discuss procedures and observations. Challenge each other. Enjoy these explorations in recursion!

A Note to the Teacher

This book is designed to be a self-teaching guided discovery manual that provides explorations into recursion. It includes a disk with lessons that are to be used along with the material in the book. Students should be sitting at the computer as they work through the lessons. The Teacher's Manual includes lesson ideas, masters for the Data Sheets, and listings of all the procedures on the disk.

Because the Logo learning environment is often self-directed and self-paced, this book is designed for individual use. It can, however, be used by small groups of students working together. If students do work together, it is important that each student completes the Data Sheet that accompanies that lesson. Masters for the Data Sheets are included in the Teacher's Manual.

Questions on the Data Sheets are numbered to correspond to the numbered steps within the lesson. When directed in the reading, students should complete the observations and exercises on the Data Sheet before continuing to the next activity. In the procedure section of each lesson, the numbered items that require students to complete a corresponding activity on the Data Sheet are marked with a $\neq$ symbol. Students also should turn in the Data Sheet for evaluation before proceeding to the next lesson.

The material in this book was designed for use on Apple II, Macintosh, or MS-DOS computers. *LogoWriter* also is required. A color monitor is highly recommended because color is used throughout the lessons.

For convenience, the procedure CS is included in each of the lessons on the accompanying disk. It may be helpful to think of it as ClearScreen because it is a quick way to clear the screen of text and graphics and return the turtle to the home position.

Students should not be in a hurry to complete the lessons. You may wish to encourage them to repeat activities until they understand the basic concepts. They should feel free to explore

their own ideas. If they are working with others, they should discuss procedures and observations. Encourage students to challenge each other.

Lesson 1. Preliminaries

In this lesson you will first perform some experiments and then examine some Logo procedures that will introduce you to recursion. Think about these ideas over the next few days.

Objectives

- To examine some recursive and nonrecursive experiences that are not computer-related
- To define recursion
- To examine simple recursion within Logo procedures

Procedure

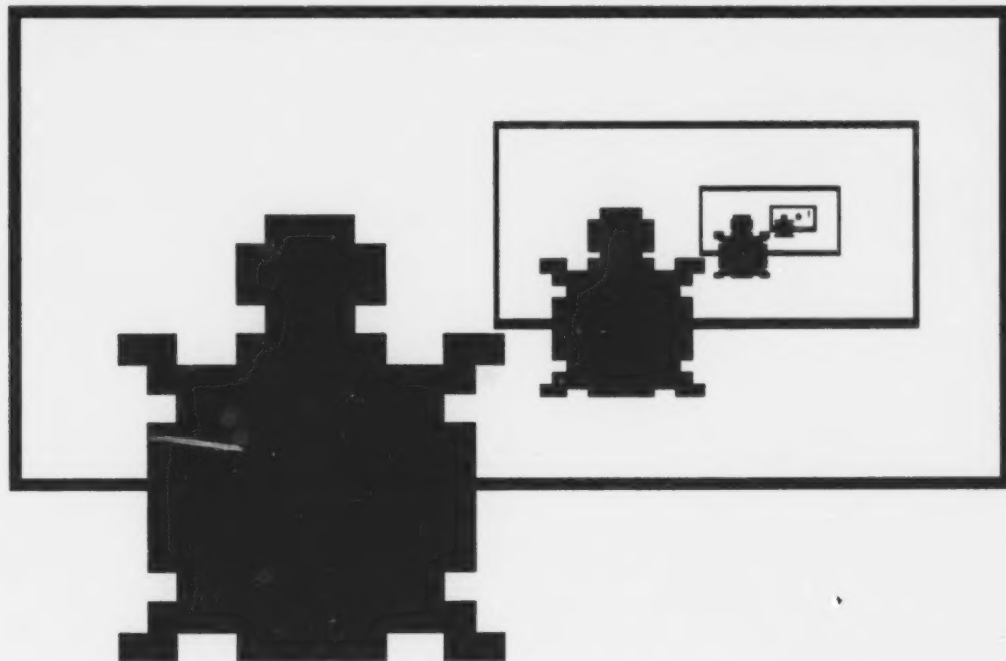
Obtain a Data Sheet from your teacher. Carry out the experiments below. Record your observations on your Data Sheet as you work through the lesson. Turn in the Data Sheet to your teacher after you finish the lesson.

- ✎ 1. Have you ever tried to call yourself? What do you think will happen? Try it! At home tonight, pick up the phone and dial the number of the phone you are using. Don't forget to write down what happens on your Data Sheet.
- ✎ 2. Take a hand mirror and stand facing a large mirror hanging on a wall. Place the smaller mirror next to your face. Looking into the large mirror, focus on the reflection of the smaller mirror. Move the smaller mirror so that you can see a part of your face and a part of the reflection in the small mirror, in the reflection of the small mirror. Can you see several reflections of the reflection in the mirror? If you don't have mirrors available in your classroom, try this at home tonight.

This is a tough experiment because of the hand-eye coordination required. You may feel a little cross-eyed when you finish.

How many small mirrors were you able to see? Record your observations on the Data Sheet.

-  3. Study the following picture from the front cover of this book. Write a brief description of the turtle-rectangle picture on your Data Sheet.



-  4. Have you ever been on a long bus ride when someone started singing "Old MacDonald"?

Old MacDonald had a farm,

Ee-i, ee-i, oh!

And on this farm he had a cow,

Ee-i, ee-i, oh!

With a "moo, moo" here and a "moo, moo" there,
Here a "moo," there a "moo," everywhere a "moo, moo,"
Old MacDonald had a farm,
Ee-i, ee-i, oh!

Old MacDonald had a farm,
Ee-i, ee-i, oh!

And on this farm he had a duck,
Ee-i, ee-i, oh!

With a "quack, quack" here and a "quack, quack" there,
Here a "quack," there a "quack," everywhere a "quack,
quack,"

"Moo, moo," here, "Moo, moo," there

Here a "moo," there a "moo,"

Everywhere a "moo, moo,"

Old MacDonald had a farm,

Ee-i, ee-i, oh!

....

Have you ever heard the song about the old lady who
swallowed a fly?

She swallowed a spider to catch the fly,
she swallowed a bird to catch the spider,
she swallowed a cat to catch the bird,
she swallowed a dog to catch the cat,
she swallowed the cat to catch the bird,
she swallowed the bird to catch the spider,
that wiggled and jiggled and tickled inside her.
She swallowed the spider to catch the fly.

I don't know why she swallowed the fly.

Perhaps she'll die.

Note your observations about these two songs on the Data Sheet.

Comments

Each of the above experiments is an example of recursion. Actually, the first experiment shows that the telephone company does not allow recursion. Have you begun to formulate a definition of recursion for yourself? See if you can put it into words.

Now go to the computer and start Logo. Use the Scrapbook disk that came with this book.

5. Load the page entitled Lesson1. On the flip side, look at the procedures called GREET and BOX. Flip back and type

GREET

in the Command Center.

-  6. Clear the text by typing

CT

In the Command Center, type the command

PRINT :NAME

Look near the top of the screen. What happened?

Now try typing

PRINT :SIDE

What happened?

Go to the flip side and study the GREET and BOX procedures. GREET uses the variable NAME and BOX uses the procedure input SIDE.

Repeat this step if you wish. Spend some time thinking about what happens. Don't forget to record your observations on the Data Sheet.

-  7. Flip to the front side. Now try this. Type

```
CS
SETC 2
RIGHT 90
DOT.LINE
```

You will need to stop the turtle. Do this with the Stop keys (see Appendix A).

Try this:

```
CS
SETC 3
RIGHT 10
DOT.LINE
```

Again, stop the turtle with the Stop keys.

To see the procedure DOT.LINE, type

```
SHOWME
```

Notice that DOT.LINE uses DOT.LINE as a subprocedure! Complete Step 7 on the Data Sheet.

Discussion

Recursion occurs when a procedure uses itself as a subprocedure. This is a deceptively simple definition for a very powerful idea. Think about it between now and the time you do the next lesson. Also, complete Step 8 on the Data Sheet before going on to the next lesson.

How did NAME acquire a value?

What happened when you typed PRINT :SIDE?

How did SIDE acquire a value?

7. Look at the procedure DOT.LINE. Explain why you need to stop the turtle.

8. If I found a lamp, and a genie appeared and granted me two wishes, my first wish would be for a shiny new car. My second wish would be for two more wishes! What would your two wishes be?

Lesson 2. Embedded Procedures

In this second lesson you will examine Logo procedures that call other procedures.

Objective

- To develop a conceptual model for embedded procedure calls that are nonrecursive

Procedure

1. Obtain a Data Sheet from your teacher. Start Logo and load the page Lesson2.

2. Type the command

```
COUNT.BOX
```

3. Go to the flip side and study the procedures COUNT.BOX and BOX. Return to the front side and type

```
CS
```

```
COUNT.BOX
```

4. Go to the flip side and position the cursor on the blank line above the END in the box procedure. Add these lines:

```
PRINT [We are now in the box procedure.]  
PRINT [Press any key to return to count.box]  
MAKE "KEY READCHAR
```

Next, position the cursor on the blank line in the COUNT.BOX procedure. Add these two lines:

```
PRINT []  
PRINT [We're in count.box now!]
```

5. Flip back to the front side. Type

```
CS
```

```
COUNT.BOX
```

6. Type the command

LEAVEPAGE

Discussion

Here's a good way to think about the COUNT.BOX procedure. The turtle relies on helpers when doing a job. Each helper is responsible for and capable of completing only one procedure.

When you type

COUNT.BOX

in the Command Center, a helper is summoned. This first helper obtains a copy of the procedure and begins to tell the turtle what needs to be done. The first instruction is a primitive, so the helper tells the turtle and the turtle carries out the instruction. The helper then continues on to the next instruction. If it is a primitive, it instructs the turtle. When this first helper encounters an instruction that is not a primitive but rather a procedure name, it must pause and call a second helper.

The second helper gets a copy of the BOX procedure. It in turn tells the turtle step-by-step what is to be done and summons helpers as needed. When the second helper is finished, it reports back to the first helper.

This first helper waits until the second helper reports back that it has finished. Only then can the first helper continue with its list of instructions.

You may want to go to the flip side and study the procedures COUNT.BOX and BOX again. Flip back and try running the COUNT.BOX procedure again.

Now complete the steps on the Data Sheet.

1. Write a procedure to draw a square.

1. Write a procedure to draw a square.
2. Write, enter, test, and debug a procedure to
 - print your first name
 - call the square procedure and draw a square
 - print the date
3. Using the idea of helpers, explain how your procedure in Step 2 works.

Lesson 3. Simple Recursion

You will begin your formal study of recursion in this lesson. First you will examine simple procedures that use recursion and then you will learn to diagram recursive procedures.

Objectives

- To explore simple procedures that use recursion
- To learn to diagram recursive procedures
- To create and diagram a recursive procedure

Procedures

Obtain a Data Sheet from your teacher. Start Logo and use the Scrapbook disk that came with this book. Record observations on your Data Sheet as you work through the lesson.

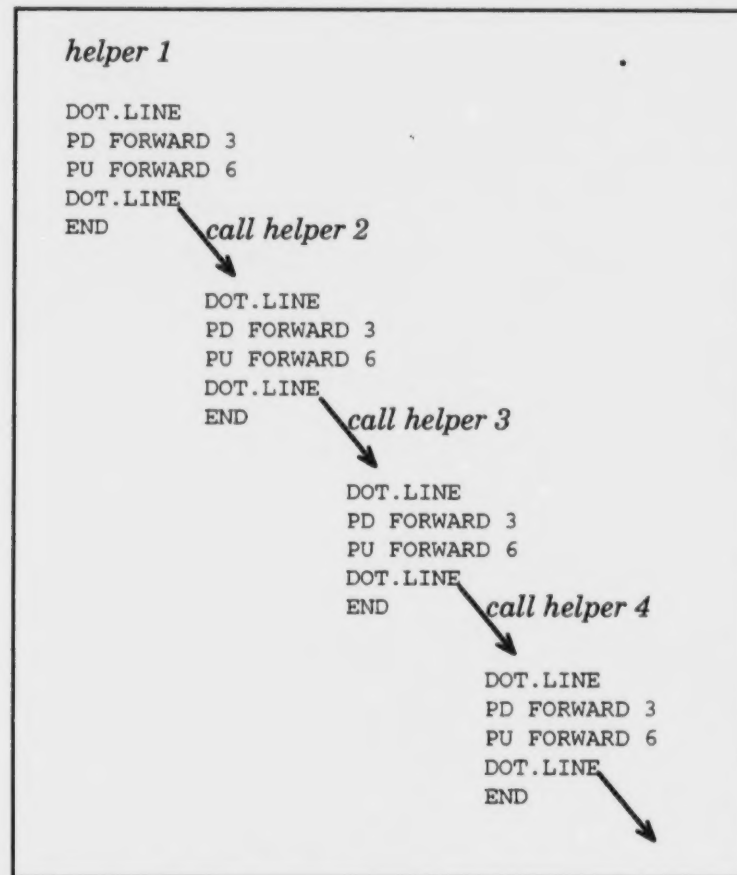
1. Load the page Lesson3 and type the following commands:

```
RIGHT 90  
SETC 2  
DOT.LINE
```

You will need to stop the turtle with the Stop keys (see Appendix A).

2. Flip to the flip side and study the procedure DOT.LINE.

The following diagram will help you understand the inner workings of this procedure.



Comments

It will be useful for you to think of Logo as having an endless supply of helpers. When a procedure contains a recursive call, a helper is called upon. This helper gets a fresh copy of the procedure and begins to carry out the instructions listed within the procedure. When reaching the recursive call instruction, the helper pauses (because it can only handle primitives contained within the procedure) and calls upon another helper.

Each helper is waiting, remembering its exact location in its copy of the procedure. When the helper that it called reports back, then this helper can continue with the steps in its procedure.

-  3. Return to the front side of the page.

To clear the screen, type the command

CS

Run the procedure DUM.DUM, by typing

DUM.DUM

You will need to stop the procedure with the Stop keys.

Study the procedure DUM.DUM on the flip side. On the Data Sheet for Lesson 3, diagram the DUM.DUM procedure as illustrated above for the DOT.LINE procedure.

-  4. Create and diagram a recursive procedure of your own. Enter it on the flip side. Test and debug it. You should have the recursive call as the last statement in your procedure. Complete the activities for Step 4 on the Data Sheet.

Data Sheet for Lesson 3

3. Diagram of the procedure DUM.DUM

4. List the commands in your recursive procedure.

Diagram your recursive procedure.

Lesson 4. Recursion Involving Procedure Inputs

In this lesson you will examine procedures that use procedure inputs along with a recursive call. You will again diagram procedures to further your understanding of helpers and recursion.

Objectives

- To investigate procedures that involve both recursion and procedure inputs
- To reinforce the concept of helpers in recursion

Procedures

Obtain a Data Sheet from your teacher. Start Logo and use the Scrapbook disk that came with this book. Record your observations on the Data Sheet.

1. Load the page Lesson4 and type the following commands:

SPIRAL 5 90

You will need to use the Stop keys to stop the turtle.

Try the spirals listed below. Type CS to clear the screen between spirals.

SPIRAL 5 60

SPIRAL 5 125

SPIRAL 5 120

SPIRAL 5 88

-  2. Study the procedure SPIRAL on the flip side. The value of SIDE controls the length of the initial line drawn. The value of ANGLE controls the size of the angle between line segments.

Try some numbers of your own.

On the Data Sheet for this lesson, record the command used to generate your favorite spiral. Also, sketch your favorite spiral (or use PRINTSCREEN).

- ✎ 3. On the Data Sheet, try to diagram the first four copies of the SPIRAL procedure. Do this now before continuing this lesson.
- ✎ 4. Once again, helpers are used when a procedure calls another procedure. To help illustrate this, the SPIRAL procedure has been altered so that each helper instructs the turtle to draw in a different color. Because we have only eight colors to choose from, the colors do repeat and occasionally the turtle draws in black.

Type CS to clear the screen between spirals. Try

```
SPI.COLOR 5 1 60
```

```
SPI.COLOR 5 1 88
```

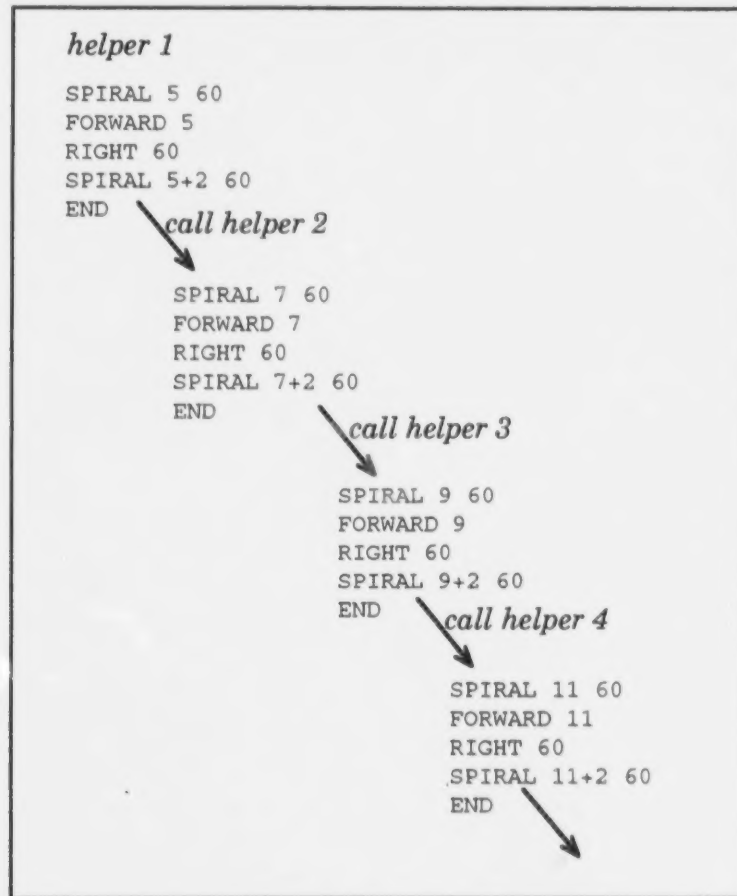
```
SPI.COLOR 5 1 120
```

Try some numbers of your own!

Record on the Data Sheet the command that generated your favorite spiral.

5. Now let's look at the diagram of the SPIRAL procedure you generated in Step 3.

Let's use a specific example, say SPIRAL 5 60.



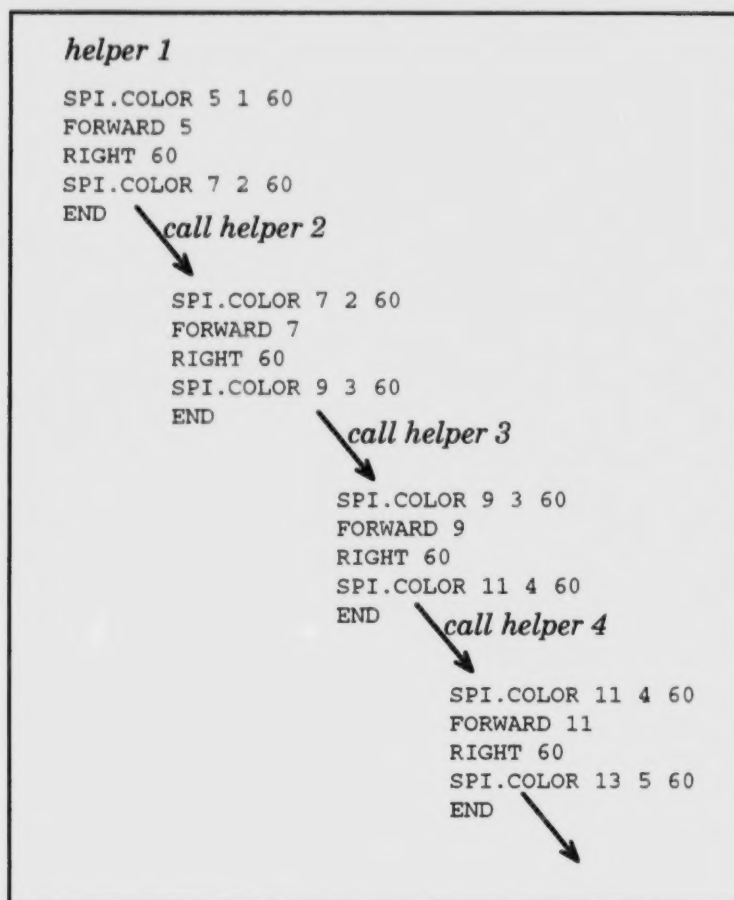
Notice that in each copy of the SPIRAL procedure, the values of the procedure inputs have been simplified and given explicitly.

Notice that the value of SIDE has increased by 2 in each copy. That is what causes the spirals to grow.

6. On the Data Sheet, try to diagram the first four copies of the procedure

SPI.COLOR 5 1 120

Use explicit values for the inputs. When you finish, compare your diagram with the following one.



7. The SPIRAL procedure has been modified once again. This time each helper will identify itself before instructing the turtle to draw. Again, different helpers will use different colors. The SPI.HELPER procedure also has been modified so that it stops by itself! You will learn how to do this in the next lesson.

Type CS between spirals to clear the screen. Then try

```
SPI.HELPER 5 1 60
SPI.HELPER 5 1 88
SPI.HELPER 5 1 120
```

Data Sheet for Lesson 4

2. Record the command that generated your favorite spiral.

Sketch your favorite spiral.

3. Diagram the first four copies of the SPIRAL procedure. You might wish to use explicit values, such as 5 and 60, for SIDE and ANGLE.

4. Record the command that generated your favorite spiral.

6. Diagram the first four copies of SPI.COLOR 5 1 120.

Lesson 5. Stop Rules

In this lesson you will learn to create and diagram procedures that stop themselves.

Objectives

- To learn to use conditionals to stop recursive procedures
- To practice diagramming procedures that use stop rules

Procedures

Obtain a Data Sheet from your teacher. Record your observations on the Data Sheet as you work through the lesson. Start Logo and use the Scrapbook disk that came with this book.

You have been working with several recursive procedures in the past lessons. Because of the recursion, you have had to use the Stop keys with each of these to stop the procedure. In this lesson, you will explore ways for procedures to stop on their own.

This lesson will continue to use the simplest form of recursion, tail recursion—with the recursive call as the last statement in the procedure. As you work through the exercises below, and as you reflect on the procedures, recall the concept of “helpers.” When a helper calls a second helper, the first helper waits until the second helper reports back that it has finished (stopped). A helper stops when the STOP command is encountered or when there are no more instructions to perform (END).

1. Load the page Lesson5 and type the following commands:
`DOT.LINE 1`
The procedure will stop itself!
Go to the flip side and study the procedure. Answer question 1 on the Data Sheet.
2. The SPIRAL procedure used in the last lesson also is included with this lesson. It has been modified with a stop

rule. Try the spirals below, along with some of your own. Remember to clear the screen between spirals with the command CS.

```
SPIRAL 10 88
```

```
SPIRAL 10 125
```

```
SPIRAL 10 170
```

```
SPIRAL 10 60
```

Study the procedure SPIRAL on the flip side. Answer question 2 on the Data Sheet.

-  3. The procedure DOT.LINE, which you have used previously, has been modified with a stop rule. Also, each helper will draw in a different color. The procedure is now DASHES; and a single input, COLOR, has been added. Try these commands:

```
CS  
DASHES 1
```

Study the procedure on the flip side. Try to diagram the procedure on the Data Sheet. Use explicit values for the procedure inputs. Check your answer with the diagram on the next page.

-  4. Next, you'll try modifying the DASHES procedure. Go to the flip side and retype the IF statement, placing it in the blank line after the recursive call statement. Then delete the IF command in the second line. Return to the front side and type

```
CS  
DASHES 1
```

What happened? Record your answer on the Data Sheet.

helper 1

DASHES 1
 IF 1 > 4 [STOP]
 SETC 1
 PD FORWARD 10
 PU FORWARD 15
 DASHES 1+1

→ *call helper 2*

DASHES 2
 IF 2 > 4 [STOP]
 SETC 2
 PD FORWARD 10
 PU FORWARD 15
 DASHES 2+1

→ *call helper 3*

DASHES 3
 IF 3 > 4 [STOP]
 SETC 3
 PD FORWARD 10
 PU FORWARD 15
 DASHES 3+1

→ *call helper 4*

DASHES 4
 IF 4 > 4 [STOP]
 SETC 4
 PD FORWARD 10
 PU FORWARD 15
 DASHES 4+1

→ *call helper 5*

DASHES 5
 IF 5 > 4 [STOP]

↙ *helper 5 stops and
reports back to helper 4*

END

↙ *helper 4 ends and
reports back to helper 3*

END

↙ *helper 3 ends and
reports back to helper 2*

END

↙ *helper 2 ends and
reports back to helper 1*

END

helper 1 ends

Discussion

The following observations can be made from the preceding examples:

- Stop rules using conditionals are an effective way to control recursive procedures that use variables.
- The stop rule is often the first instruction in a recursive procedure.
- The stop rule **MUST** appear before the recursive call.

Data Sheet for Lesson 5

1. Study the procedure DOT.LINE. Which line contains the recursive command?

The conditional (IF statement) is used to stop the recursion. Which line of the procedure contains the IF statement?

2. Study the procedure SPIRAL. Which line contains the recursive call?

Which line contains the IF statement?

The SPIRAL procedure uses two variables, SIDE and ANGLE. Which one is used in the stop rule?

Which procedure input is increased in the recursive call?

3. Diagram the DASHES procedure. Use explicit values for the procedure inputs.

4. What happened when you modified the DASHES procedure?

Lesson 6. Embedded Recursion

So far, you have examined procedures that use tail recursion. In this lesson you will be introduced to procedures with an embedded recursive call.

Objectives

- To explore procedures that use embedded recursion
- To predict the results of procedures that use embedded recursion
- To explore patterns and processes within procedures

Procedures

Obtain a Data Sheet from your teacher. Record your observations on the Data Sheet as you work through the lesson. Start Logo and use the Scrapbook disk that came with this book.

All of the recursive procedures you have studied thus far have had the recursive call as the last statement before the END statement in the procedure. It is not necessary that the recursive call be placed at the end. In fact, many interesting results can be obtained by using embedded recursion. You will want to recall our earlier discussions about helpers as you work through this lesson. In particular, you will want to remember that when a helper calls on another helper, the first helper waits until the second helper reports back that it has finished. The first helper can then continue carrying out instructions in the procedure.

-  1. Load the page entitled Lesson6.

The SPI.HELPER procedure from the last lesson has been modified by placing the recursive call near the middle of the procedure. There are instructions to be carried out after the recursive call. Try the procedure by typing

```
SPI.HELPER 10 1 60
```

Study the procedure on the flip side. There are instructions to be carried out after the recursive call. Clear the screen (CS) and try the procedure again. Notice that each helper identifies itself as it works. Answer the questions for this step on the Data Sheet.

-  2. Two short procedures have been included to move and draw a square. One draws a "right square" and the other draws a "left square." Try each one. Type

```
CS
SQR 1
```

```
CS
SQL 1
```

Answer the questions on the Data Sheet.

-  3. The two procedures above are used in the procedure listed below.

```
TO SQUARES :COLOR
IF COLOR > 4 [STOP]
SQR :COLOR
SQUARES :COLOR + 1
SQL :COLOR
END
```

Do not run this procedure yet. Instead, go to the Data Sheet and answer the questions there.

-  4. Run the procedure SQUARES by entering the command

```
CS
SQUARES 1
```

Compare the results with your prediction. Record your observations on the Data Sheet.

Below is a diagram of the procedure, indicating the passage of time as you move down the page. Study the diagram. Be sure you understand what each helper is doing as time progresses.

helper 1

SQUARES 1
 IF 1 > 4 [STOP]
 SQR 1
 SQUARES 1+1

call helper 2

SQUARES 2
 IF 2 > 4 [STOP]
 SQR 2
 SQUARES 2+1

call helper 3

SQUARES 3
 IF 3 > 4 [STOP]
 SQR 3
 SQUARES 3+1

call helper 4

SQUARES 4
 IF 4 > 4 [STOP]
 SQR 4
 SQUARES 4+1

call helper 5

SQUARES 5
 IF 5 > 4 [STOP]

*helper 5 stops and
 reports back to helper 4*

SQL 4
 END

*helper 4 ends and
 reports back to helper 3*

SQL 3
 END

*helper 3 ends and
 reports back to helper 2*

SQL 2
 END

*helper 2 ends and
 reports back to helper 1*

SQL 1
 END

helper 1 ends

- ✎ 5. Let's try another procedure. Below are the commands for a FANCY.NAME procedure:

```
TO FANCY.NAME :NAME  
IF :NAME = "[STOP]  
PRINT :NAME  
FANCY.NAME BUTFIRST :NAME  
PRINT :NAME  
END
```

Do not run the procedure yet. Instead, try to predict what will happen when you type the procedure name followed by your first name.

Record your predictions on the Data Sheet. Diagramming the procedure may help you predict the outcome.

If necessary, type CS to clear the screen. Now run the procedure, and don't forget to type your name preceded by double quotes ("). Compare the outcome with your prediction.

- ✎ 6. Below are the results from two different but similar procedures. Each procedure uses the same base shape. Can you identify it? Record your answer on the Data Sheet.

(a)



(b)



Did you select a V as the base shape, with the left side twice as long as the right side? See if you can find several of these shapes in each of the pictures. Did you notice that the size of the Vs gets smaller and smaller?

One of the pictures was created with a procedure that uses tail recursion (recursive call as the last statement), while the other uses embedded recursion. Identify each on the data sheet.

Data Sheet for Lesson 6

1. Study the procedure SPI.HELPER. Which line contains the recursive command?

What commands are carried out before the recursive call?

What commands are carried out after the recursive call?

2. After typing SQR 1, what is the heading of the turtle?

What is its position relative to its starting position?

After typing SQL 1, what is the heading of the turtle?

What is its position relative to its starting position?

3. Try to predict what the outcome of this procedure will be.
Below, draw what you think the turtle will draw.

You may want to diagram the procedure if you are having trouble predicting the outcome.

4. Compare your predicted outcome with the actual outcome of SQUARES.

5. Predicted outcome of the FANCY.NAME procedure

6. Sketch one copy of the base shape below:

Which picture was created with tail recursion?

Which picture was created using embedded recursion?

Lesson 7. Logo Variables

In this lesson you will return to some ideas introduced in the first lesson. You have been using procedure inputs in many of the recursive procedures. You'll explore their nature and also will learn about global variables in this lesson.

Objective

- To explore Logo variables

Procedures

Obtain a Data Sheet from your teacher. Record your observations on the Data Sheet as you work through the lesson. Start Logo and use the Scrapbook disk that came with this book.

In Lesson 1 we looked at two procedures, GREET and BOX. GREET uses a variable called NAME and BOX uses a procedure input called SIDE.

1. Load Lesson7 from the disk that came with this book. Type the command

GREET

Clear the text by typing

CT

In the Command Center, give the command

PRINT :NAME

PRINT :SIDE

Did the turtle respond by printing your name at the top of the screen and by complaining in the Command Center that SIDE has no value?

Comments

If you have studied any programming languages other than Logo, you have encountered the idea of variables. You also have

encountered variables in Logo. Often, variables are explained by comparing them to containers or to mail boxes.

The computer's memory is often compared to the wall of postal boxes at the post office. There is one very important difference, however. Each box can hold only one piece of information at a time. In BASIC, we would say that each box can hold only one string value or one numeric value at a time. When a new value is put into the box, the old value is gone forever. In Logo we would say that each variable can hold only one object (a word or a list) at a time.

In Step 1 above you noticed that the variables NAME and SIDE behave differently after the procedure has finished executing. That is because there are two types of variables in Logo. In Lesson 1, you were asked on the Data Sheet to explain how NAME and how SIDE were given their initial values.

-  2. Go the flip side and study the procedures GREET (which calls GETNAME) and BOX. How are NAME and SIDE given their initial values? Record your comments on the Data Sheet now.

Comments

The variable NAME is an example of a global variable. Global variables get their values from MAKE or NAME statements. Once global variables are given a value, they are available to all procedures. In fact, if another page is loaded, the previously used global variables are available to it. You noticed that global variables retain their value after the procedure that gave them values have finished executing. These global variables in Logo behave much the same as the variables in BASIC.

-  3. Flip to the front side and type

DAYNAME

In the Command Center you are asked

What day is it?

Type Monday. Again you will be asked what day it is. Type Tuesday. The process continues—enter Wednesday, Thursday, Friday, and Saturday.

What do you notice? Pay particular attention to the helper number and how it changes and to the day name and how it changes. Run the procedure again, if necessary. Record your observations on the Data Sheet.

Comments

The DAYNAME procedure uses two variables, DAY and HELPER. It calls GETDAY and NAME.DAY. GETDAY gives DAY a value with the MAKE command.

The NAME.DAY procedure uses the variable HELPER, which is a local variable. Local variables have values passed to them through procedure calls. The values of these variables are gone after the procedure finishes executing.

In the NAME.DAY procedure, instructions to print the values of DAY and HELPER are given before and after the recursive call.

4. On the flip side, study the procedure GETDAY. Recall how the helper number changes as the procedure executes. Reflect on this for a few minutes. Rerun the procedure if necessary. Go to the front side and run DAYNAME once more. When it finishes, enter the commands

```
PRINT :DAY  
PRINT :HELPER
```

Record your thoughts on the Data Sheet.

Comments

Unlike global variables, local variables can hold more than one value at a time. We will think of local variables as containers that can hold many things in an orderly way. Only the last value placed in the container is available, and it is available to the helper that is currently active. The last value to be placed in the container is on top and is available for use, and the first thing that was placed there by the first helper is on the bottom.

When a value is passed to a helper, that value is placed on the top in the container. When the helper finishes its task, it removes its value from the top of the container.

Step 5 below will illustrate how local variables work. In Step 5 you will run a procedure called MEMORY. To help you understand local variables, a container labeled COLOR will be drawn in the lower left corner of the screen. Five different helpers

(four on MS-DOS computers) will be required during the running of this procedure. They will identify themselves by number while they are working. The value of the local variable COLOR (which equals the helper number) will be placed in the container at the time the helper is called. After identifying itself, the helper draws a square. It then needs to summon another helper (the recursive call), and it waits until that helper reports back that it has finished. It then is able to finish, and reports back to the helper that called it, removing its value of COLOR.

5. On the front side, enter the commands

```
CS  
MEMORY
```

Notice the values of COLOR in the container and how they change as the different helpers work.

-  6. On the Data Sheet write a few paragraphs explaining your understanding of global and local variables.

Data Sheet for Lesson 7

2. How are NAME and SIDE given their initial values?

3. In the DAYNAME procedure, how does the helper number change?

How does the day name change?

4. Reflections on the DAYNAME procedure

6. Explain global and local variables.

Lesson 8. Introduction to Fractals

In this lesson you will investigate an application of recursion—fractals. Some of the material in this lesson appeared in the article “Fractals I: Making Recursion Visible,” by Mark Horney, in the September 1990 issue of *Logo Exchange*.

Objectives

- To provide an introduction to the exciting world of fractals. Writing fractal procedures requires a good understanding of recursion.
- To motivate you to continue your Logo explorations

Procedures

Obtain a Data Sheet from your teacher. Record your observations on the Data Sheet as you work through the lesson. Start Logo and use the Scrapbook disk that came with this book. Load the page entitled Lesson8.

You have been doing a lot of work with recursion and have observed many interesting results. An exciting new area of mathematics that has received a lot of attention lately is fractals. Fractals rely on recursion in creating their magic. The purpose of this lesson is to spark your interest in fractals. Your teacher can supply reading material and more ideas for exploration if you are interested in further work with fractals.

Let's begin by looking at a few fractals.

1. In the Command Center, enter the command

POSITION

This just moves the turtle down on the screen so that the design fits better. Enter the command

DRAGONSKIN 100 5

It might take the turtle a while to draw the design. Also, the turtle may seem to turn from side to side without purpose.

2. Let's examine another fractal. Enter the commands

```
POSITION  
SNOWFLAKE 80 3
```

Study the design for a few minutes.

We can build a full snowflake with the following commands:

```
POSITION  
REPEAT 6 [SNOWFLAKE 80 3 LEFT 60]
```

Study this design for a few minutes.

3. Next, try this:

```
POSITION.SPIKE  
SPIKE 200 5
```

Again study the design before continuing to the next step.

4. Enter the commands

```
POSITION.SQFLK  
SQUAREFLAKE 260 4
```

- ✍ 5. Reflect on each of the four fractals and write your observations on the Data Sheet.
- ✍ 6. Flip to the flip side and look briefly at the procedures that define the fractals. Do you see recursive calls? Each of the procedures uses two variables, SIDE and LEVEL. On the front side, experiment with creating other fractals by first using different level numbers. Record your observations on the Data Sheet.
- ✍ 7. Next, experiment with creating other fractals by changing the SIDE but keeping LEVEL at 4. Record your observations on the Data Sheet.
8. There are many concepts to be explored about fractals. Some of these are listed below.
- How many turtle steps does the turtle take in drawing a fractal? (What is the perimeter of the fractal?) Think about this question for different levels of the same fractal.
 - What happens to the perimeter as the number of levels gets very large? (Don't try this on the computer.)
 - If you were to time the turtle when drawing the fractal and use your measure of perimeter, you could calculate the

turtle's velocity. Will the velocity change when the number of levels changes? Will a change in the length of the side change the turtle's velocity?

- What is the area enclosed by the full snowflake fractal? How does it change as the number of levels increases (keep SIDE the same)?
 - How many procedure calls are made by the squareflake fractal when the level is 2? 3? 4?
 - If you decide to investigate any of these questions, you may want to modify the procedures to calculate the perimeter and count recursive calls for you.
9. All of the fractals included in Lesson 8 are examples of Koch curves, named for the mathematician Helge von Koch, who investigated them.

Here is a technique Mark Horney gives in the article mentioned at the beginning of this chapter for drawing Koch fractals.

Step 1. Start with a shape, usually a line segment, called the "initiator." (The snowflake uses a triangular initiator; the others use a line segment.)

Step 2. Replace each piece of the initiator with another series of shorter line segments, called the "generator."

Step 3. Treat each segment of the generator as if it were an initiator and replace them with reduced copies of the generator. Repeat this process for as many "levels" as your patience and computing power allow.

Study the procedure listing on the flip side for your favorite fractal. Can you see the elements described above in the listing for the fractal?

After experimenting with the four fractals provided, you may want to experiment with drawing your own.

Discussion

The purpose of the lesson was to present an example from mathematics where recursion is used. Several interesting questions also were presented. If you would like to study fractals in more detail, please see your teacher. Your teacher can provide materials for your further explorations.

Data Sheet for Lesson 8

5. Record your reflections on the four fractals.

6. Observations on changes in LEVEL

7. Observations on changes in SIDE

Appendix A. LogoWriter Keys

General-Purpose Keys

	APPLE	IBM	MACINTOSH
Flip Flips a page over	Apple-F	Ctrl-F	⌘-F
Up Moves the cursor up from Command Center	Apple-U	Ctrl-U	⌘-U
Down Moves cursor down into the Command Center	Apple-D	Ctrl-D	⌘-D
Esc Used to leave a page	Esc	Esc	Esc or Clear
Stop Stops program or instruction; returns to Command Center (see also Esc)	Apple-S	Ctrl-Break	⌘-.
Help Gets the Help page	Apple-0	Fn-10	—

Contents Page Keys

	APPLE	IBM	MACINTOSH
Up and Down Arrows Moves cursor up and down through pages that are in the scrapbook	Up & Down Arrows	Up & Down Arrows	Up & Down Arrows
Top of Page Moves the cursor to the top of the Contents list	Apple-Up Arrow	Home	—
Bottom of Page Moves the cursor to the last page name in the Contents list	Apple-Down Arrow	End	—
Return/Enter Chooses the page that the cursor is on	Return	Enter	Return
Erase to End of Line Erases the page named on the line that the cursor is on—permanently!	Apple-6	Fn-6	—

Graphics Keys

	APPLE	IBM	MACINTOSH
Turtle-Move Makes it possible to move the turtle with the Arrow keys; Esc to leave	Apple-9	Fn-9	⌘-9
Label Text typed is added to the picture; use Arrow keys to move; Esc to leave	Apple-8	Fn-8	⌘-8

Put the Future of Education in Your Hands...



Pick up David Thornburg's visionary books that explore the social and technological trends shaping education.

Edutrends 2010—Restructuring, Technology, and the Future of Education

Thornburg examines the recent flurry of teacher bashing, moving from the reasoning flaws of those who would dismantle U.S. public education to the change process necessary for educational restructuring for the 21st century.

He proposes a way to think intelligently about the future while we reshape education to meet the needs of our society.

Education, Technology, and Paradigms of Change for the 21st Century

Learn unique patterns of thinking that encourage students to go above and beyond the three Rs. The technologies of calculators, computers, laser discs, and video games converge with new concepts of learning and intelligence to provide unprecedented opportunities for all students. Let this progressive book lead you toward a future filled with new paradigms.



International Society for Technology in Education
1787 Agate Street, Eugene, OR 97403-1923
Order Desk: 800/336-5191 Fax: 503/346-5890

All you need is the right tool...

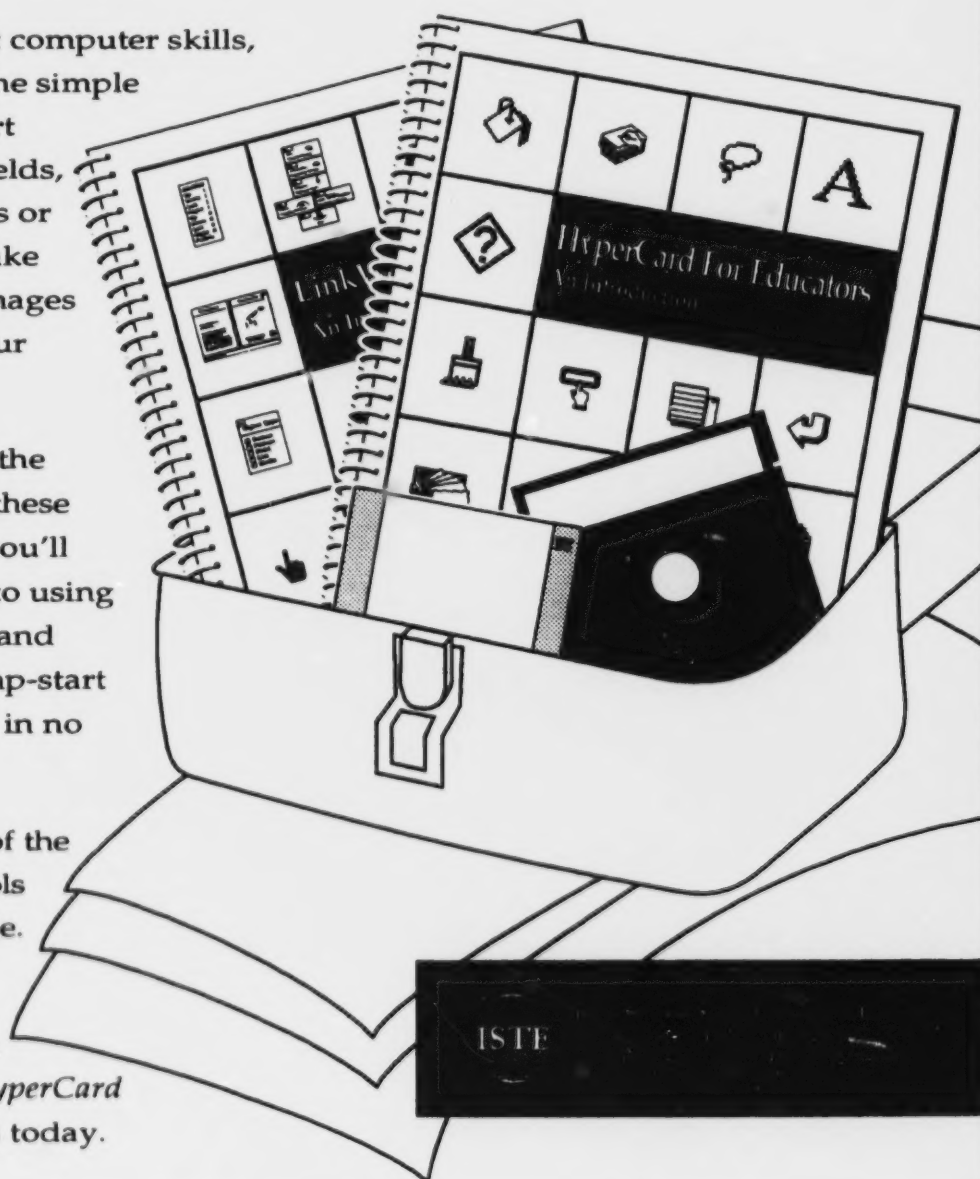
With either *HyperCard for Educators* (Macintosh) or *LinkWay for Educators* (MS-DOS) as an introduction to the interactive world of hypermedia, you can learn to build libraries of knowledge suitable for both classroom and personal use.

If you have basic computer skills, you can follow the simple tutorials and start using buttons, fields, and backgrounds or base pages to make your data and images come alive in your classroom.

By starting with the foundation that these books provide, you'll be on your way to using sounds, images, and animation to jump-start your curriculum in no time!

Take advantage of the organizational tools computers provide.

Call ISTE to order and begin building your *HyperCard* or *LinkWay* skills today.



Make This Step Your Next Step.

You've got your new Mac...
Now what?

You want to know what those cute icons mean. You need to find out how to use that mouse-thing-a-ma-jig.

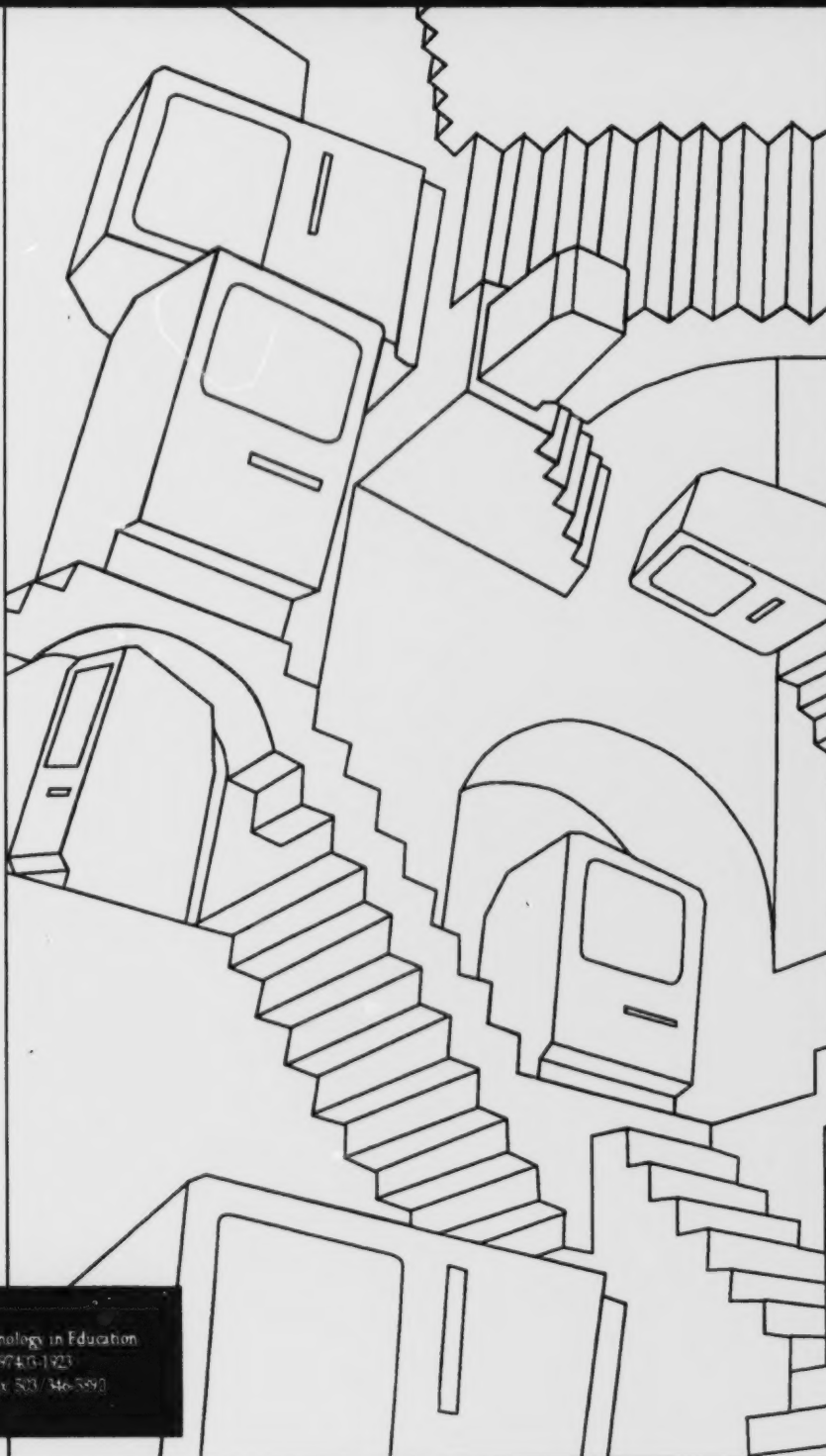
You'd like to know how to "empty the trash," and it'd be great to understand exactly what the "chooser" chooses.

By following the footsteps in ISTE's new book, *Macintosh Step by Step*, you'll pick up on all these basic Macintosh operating procedures, plus more—from finding the "power" button to demystifying the little Apple menu in the upper left corner.

And it also teaches you the theory behind Mac file storage and the importance of backing up your data on a regular basis.

In all, you'll find that this simple, illustrated guide will lead you on your way toward becoming a Mac user—not just a Mac owner.

Just call today and make your next computing step a step in the ISTE direction.



ISTE

International Society for Technology in Education
1787 Agate Street, Eugene, OR 97403-1923
Order Desk 800/336-5191 Fax 503/346-5890

ISTE Books & Courseware Order Form

Name _____

School/Business _____

Address _____

City _____ State _____ Zip/Postal Code _____

Country _____ Phone _____

☐ Yes! I'm presently an ISTE member. Membership number must be included to receive discount.
My membership number is _____ (Discount does not apply to initial membership fee.)

Occupation

Grade Level: ☐ all (K-12) Specific grade level _____

☐ Teacher (350) Subject _____

☐ Administrator (320) Title/Position _____

☐ Higher Education (360) Title/Position _____

☐ Federal/State Government (310) Title/Position _____

☐ Educational Information Resource Manager (IRM) (320)

☐ Technology Coordinator (340)

☐ Other _____

Payment Options

- ☐ **Payment enclosed.** (Make checks out to ISTE.) Non-U.S. orders must be pre-paid with U.S. funds or credit card.
☐ VISA ☐ MasterCard ☐ Discover Card Exp. Date _____
☐ **Purchase Order enclosed.** (Please add \$2.50 for order processing. P.O. not including \$2.50 fee will be returned.)
☐ **C.O.D. for U.S. Book orders only.** You will pay UPS the total upon delivery (check or cash—I STE will add \$2.75 order processing).

Terms & Conditions

Terms: No 60 orders must be accompanied by authorized company purchase orders. Orders for individuals must be pre-paid, C.O.D., or charged to a Visa, MasterCard, or Discover card number. All non-U.S. orders must be pre-paid.

Shipping: Shipping method is UPS. Add 5% in addition to shipping rate for first class delivery. *Allow three weeks for delivery.* Call for rush orders.

Prices: Subject to change without notice.

Returns: ISTE allows return of merchandise, provided the materials are in resellable condition. There will be a 10% restocking charge. All returns must be accompanied by an ISTE Return Authorization form. If you wish to return merchandise, please contact the Distribution Dept. GST Registration #128828431

**Shipping & Handling
for Books & Courseware**

\$0-\$15.99 (subtotal)	add \$3.25
\$16-\$45.99 (subtotal)	\$4.50
\$46-\$75.99 (subtotal)	\$5.50
\$76-\$100.99 (subtotal)	\$6.50
\$101 or more	7% of subtotal

Do not include additional site license fees or subscription costs when computing shipping rates.

Non-U.S. orders for Books & Courseware are sent surface mail. If you want your Books & Courseware order shipped AIR-MAIL, please check here. ☐ ESTE will bill you the additional shipping charge.

GST Registration Number 128828431

ISTE Books, Courseware, & Nonmember Subscriptions

[illegible]

MAIL this form to:

ISTE, 1787 Agate St., Eugene, OR
97403-1923 USA

FAX this form with credit card or P.O. information to: 503/346-5890

PHONE your credit card or P.O. order to:
ISTE Order Desk: 800/336-5191

BOOKS & COURSEWARE SUBTOTAL

Add Shipping, based on Books & Courseware SUBTOTAL

Add \$10 S & H for Don't Copy that Floppy

Add Additional 5% of SUBTOTAL if shipped to PO Box, AK, HI, or outside U.S.

Add 7% of SUBTOTAL for GST if shipped to Canada

If billed with purchase order, add \$2.50; If COD, add \$2.75

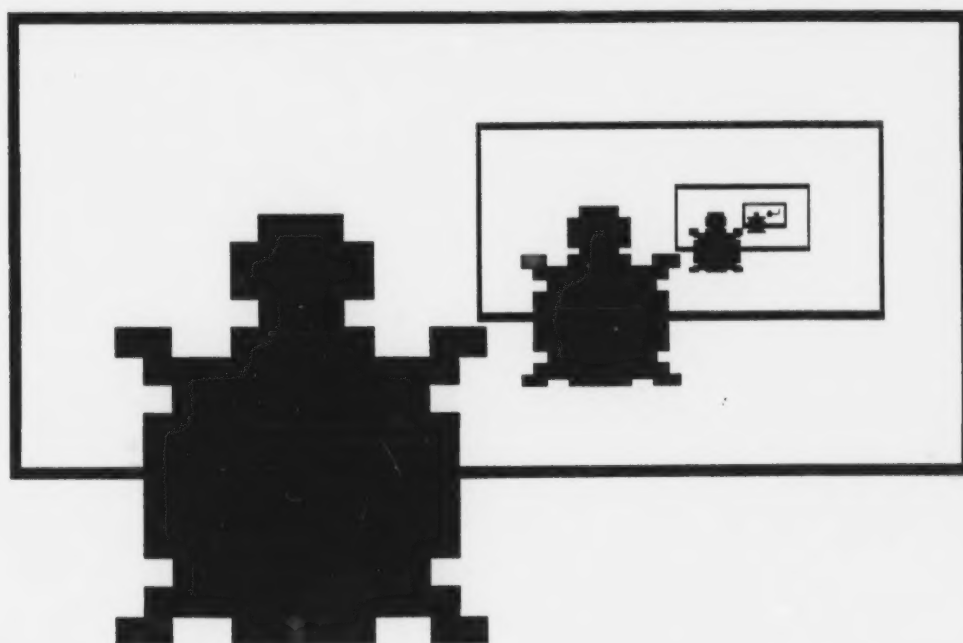
Membership Total from other side

TOTAL

Exploring Recursion

With *LogoWriter*

Teacher's Manual



Mary Salter

BEST COPY AVAILABLE

About the Author

Mary Salter received her M.S. degree in math from Loyola University in New Orleans. She has taught mathematics and computer science at high school and college levels for 15 years and is currently an associate professor of mathematics and computer science at Franciscan University of Steubenville in Steubenville, Ohio. She teaches computer programming and applications courses for preservice teachers and graduate education computer courses for teachers and administrators. She has been a contributor to *Logo Exchange* and has coauthored a text on linear algebra experiments using the DERIVE program.

Technical Editor: Sharon Yoder

Editing and Production Assistance: The Electronic Page

Book Cover: Sharon Yoder

Book Design: Jean Hall

© International Society for Technology in Education, 1993

1787 Agate Street, Eugene, Oregon 97403-1923

Order Desk: 800/336-5191

Phone: 503/346-4414

Fax: 503/346-5890

Compuserve: 70014,2117

Internet: ISTE@Oregon.uoregon.edu

ISBN 1-56484-032-8

Exploring Recursion

With *LogoWriter*

Teacher's Manual

Mary Salter

© International Society for Technology in Education, 1993

From the Publisher

The purchaser of the *Exploring Recursion With LogoWriter* package receives this book, a data disk, and a single-user site license. The *LogoWriter* application is *not* included and *is* necessary for using the disk files.

The single-user site license gives the one person who is designated as the "primary user" of the *Exploring Recursion With LogoWriter* package the right to make copies of Data Sheets, disk files, and other material necessary for use in the classes he or she teaches. The right to copy materials from the book or the disk does not transfer to participants in classes taught by the primary user. If the single-user site license is purchased by a school or school district, the intent is that one person be designated as the "primary user." If several people are to use the book for teaching, separate packages of *Exploring Recursion With LogoWriter* must be purchased for each individual teacher.

Table of Contents

Part 1. Instructions for Using This Manual 1

Part 2. Resources and Suggestions 5

Part 3. Data Sheet Masters 9

Part 4. Procedure Trees and Listings 29

Appendix A. LogoWriter Keys 43

Appendix B. Suggested Readings 45

Part 1. Instructions for Using This Manual

The Logo learning environment is often self-directed and self-paced. Thus, it may not be desirable to introduce recursion to the whole class at the same time. The purpose of *Exploring Recursion With LogoWriter* is to provide material for individual study of recursion using *LogoWriter*. It is appropriate for junior high, high school, and college students, as well as for teachers who wish to learn about recursion.

Students should be thoroughly familiar with the commands used in turtle graphics, with IF and IFELSE, with writing procedures, and with using/passing variables. Appendix A provides a chart of *LogoWriter* keys for quick reference.

Although the material is designed for individual use, it could be used effectively with a small group or even with a whole class (if all members are ready to learn recursion). Whether students are working individually or in small groups, they should work at their own pace. Students may tend to work too quickly through the material; therefore, teachers may want to encourage careful reading and reflection. The time spent on each lesson will vary from lesson to lesson and from student to student. It is more important for students to think about and comprehend the ideas of recursion than for them to finish in a specified length of time.

Exploring Recursion With LogoWriter—Teacher's Manual is designed for teachers who have students who will be using the material in the book. The teacher's role is that of a resource person who should encourage students to discover the answers to their own questions. Students often need to be encouraged to read and reread explanations and to rephrase them in their own words.

There are times, however, when students need a specific answer. This teacher's manual contains references to readings that will give the teacher further insight into recursion and the teaching of recursion. It also contains references to activities that can be used with a small group or class but that are not appropriate for individuals.

It is recommended that, as time permits, teacher's read all of the references listed in Appendix B of this book. Start with the following two articles:

- Clements, Douglas. (December 1989). "Recurrent Recursion Misconceptions." *Logo Exchange*, 8(4), 31-32
- Lough, Tom. (December/January 1983-84). "A Cure for Recursion." *The Computing Teacher*, 11(5), 34-37

You may want to read these before you begin working with students on the explorations.

Students may enjoy working on the explorations in groups of two or three. However, each student should fill out his or her own Data Sheets. A major part of the learning process involves recording observations and explaining the understanding of concepts. Masters for the Data Sheets appear at the end of this manual. See the section entitled From the Publisher at the front of this book for guidelines on reproducing the Data Sheets.

Assessment is a major concern for many teachers. Data Sheets should be reviewed before students proceed to the next lesson. There are no answer sheets included in this manual because there are no specific correct answers. When teachers review a student's work in Logo, they are looking at the student's thinking and understanding. It is an opportunity for the teacher to note the student's learning style and communication skills. It may be helpful to read over the Data Sheet and place it in one of three categories: missed the point, adequate understanding, or unusual insight or understanding. Those who missed the point will need to repeat the lesson before continuing. The Data Sheet may help identify any misconceptions. Perhaps the student just hurried through the lesson and needs to be encouraged to be more reflective. Those with an adequate understanding may proceed; however, it may be useful for the student to discuss the content of the lesson with a fellow student or with the teacher to further the understanding of the concepts. Those with an unusually good understanding should be encouraged to share it with others.

The last part of this teacher's manual includes procedure trees and listings of all the procedures contained on the disk accompanying the *Exploring Recursion With LogoWriter* package. You can modify these procedures as necessary. See the section entitled From the Publisher at the front of this book for guidelines on copying the disk files.

Part 2. Resources and Suggestions

Lesson 1. Preliminaries

The following three articles by Tim Riordon from *The Computing Teacher* suggest many activities for the teaching of recursion.

(December/January 1983-84). "Helping Students With Recursion: Teaching Strategies." *The Computing Teacher*, 11(5), 38-40

(February 1984). "Helping Students With Recursion: Teaching Strategies. Part II: Moving to the Computer." *The Computing Teacher*, 11(7), 59-64

(March 1984). "Helping Students With Recursion: Teaching Strategies. Part III: Teaching Students About Embedded Recursion." *The Computing Teacher*, 11(8), 64-69

Activities 2, 3, and 5 in the first of these three articles are appropriate for small groups or the whole class. The POW game described in Activity 6 of the second article could be quite fun with a group. Other useful activities described in Riordan's articles are mentioned in the following sections for specific lessons.

Creative teachers might want to design their own activities. There are several good examples of recursion in advertising material. A Land O Lakes butter carton and the label on a Morton Salt box can be used to demonstrate recursion in a visual context. The Indain maiden on the Land O Lakes carton holds a carton with an Indian maid holding a carton, and so on. Likewise, the Morton Salt girl holds a box of salt with a girl holding a box of salt.

Lesson 2. Embedded Procedures

If students are working together in pairs or in small groups, you might encourage one to write a procedure that calls another procedure. Have the other students verify that the goal has been

achieved. Repeat the process until each member of the group has created a procedure that calls another procedure.

In most of the lessons, students probably will encounter some unfamiliar words. You may want to preview the lesson and discuss new vocabulary before students begin work. In particular, students will encounter the following terms in Lesson 2: *embedded*, *procedure calls*, and *primitives*. It is important that they understand these terms.

Lesson 3. Simple Recursion

Encourage students working in groups to discuss the idea of helpers.

You might have one student write a recursive procedure and have another diagram it. The process should be repeated until all students have written a recursive procedure. This also will provide practice in diagramming procedures.

In this lesson, students will encounter the new phrase *recursive call*. Students should come to understand this term as they work through the lesson.

Lesson 4. Recursion Involving Procedure Inputs

The teacher might present the students with the listing of a recursive procedure that involves procedure inputs and have the students predict what will happen. If the students have trouble predicting the results, they should diagram the procedure and try again to predict the result. After the students feel confident that they have correctly predicted the outcome of the procedure, they should enter it on the computer and compare the result to their prediction.

If their prediction was not correct or if they feel they need more practice, the teacher should provide other such procedures.

Students could be assigned to work in pairs to create a recursive procedure that uses procedure inputs. They should design, debug, and diagram their procedure.

The new terms students will encounter in Lesson 4 include *procedure inputs* and *copies*. Students should be very familiar with Logo before working on recursion. It is assumed that students have previously used procedures that require inputs. The term *copies* is tied to the idea of *helpers*. When a helper is called, it obtains a *copy* of the procedure. In Procedure 3, students are asked to diagram the first four copies of the SPIRAL

procedure. Have them study the diagram in Lesson 3 if they have trouble with this activity.

Lesson 5. Stop Rules

An activity suitable for a group of students is Activity 12 in Riordan's article, "Helping Students With Recursion: Teaching Strategies. Part II: Moving to the Computer." Students could be asked to rewrite previous recursive procedures so that they include a stop rule.

As a special challenge for the more advanced students, they could be shown output and then be asked to write a procedure that creates the output.

In Lesson 5, students will encounter the terms *stop rule* and *conditionals*. Both terms refer to the IF primitive. A stop rule specifically uses an IF to STOP a procedure. Students also should develop an understanding of the term *tail recursion* as they work through the lesson.

Lesson 6. Embedded Recursion

Activities 13, 14, 15, and 17 in Riordan's article "Helping Students With Recursion: Teaching Strategies. Part III: Teaching Students About Embedded Recursion" are excellent supplementary activities for learning about embedded recursion.

Lesson 7. Logo Variables

An alternate method for discussing local variables and their values during execution is presented in Chapter 5 of Brian Harvey's *Computer Science Logo Style. Volume 1: Intermediate Programming* (see the listing in the Recursion section of Appendix B).

Teachers who prefer this description should instruct students to skip Lesson 7.

In this lesson, students will encounter the following new terms: *procedure input*, *variables*, *global*, and *local*. They should develop an understanding of these terms as they work through the lesson.

Lesson 8. Introduction to Fractals

The three articles by Mark Horney cited in the Fractals section of Appendix B give a good introduction to Koch curves. These articles are suitable for use by students and teachers.

New terms in Lesson 8 include *fractals*, *perimeter*, *velocity*, and *area*. If students ask for a definition of a fractal, you might say that a fractal is a picture that contains smaller and smaller copies of itself.

Part 3. Data Sheet Masters

These Data Sheets match those provided in each copy of *Exploring Recursion With LogoWriter*. Teachers may wish to photocopy and distribute these Data Sheets so that the students' books remain complete. See the section entitled From the Publisher at the beginning of this book for guidelines on reproducing the Data Sheets.

Data Sheet for Lesson 1

1. Observations on recursion and the telephone company
2. Observations on the mirrors experiment
3. Comments on the cover picture
4. Comments on the two songs
5. Comments on the two songs
6. What happened when you typed `PRINT :NAME?`

How did NAME acquire a value?

What happened when you typed PRINT :SIDE?

How did SIDE acquire a value?

7. Look at the procedure DOT.LINE. Explain why you need to stop the turtle.
8. If I found a lamp, and a genie appeared and granted me two wishes, my first wish would be for a shiny new car. My second wish would be for two more wishes! What would your two wishes be?

Name: _____

Data Sheet Masters

Data Sheet for Lesson 3

3. Diagram of the procedure DUM.DUM

4. List the commands in your recursive procedure.

Diagram your recursive procedure.

Data Sheet for Lesson 4

2. Record the command that generated your favorite spiral.

Sketch your favorite spiral.

3. Diagram the first four copies of the SPIRAL procedure. You might wish to use explicit values, such as 5 and 60, for SIDE and ANGLE.

4. Record the command that generated your favorite spiral.

6. Diagram the first four copies of SPI.COLOR 5 1 120.

Data Sheet for Lesson 5

1. Study the procedure DOT.LINE. Which line contains the recursive command?

The conditional (IF statement) is used to stop the recursion. Which line of the procedure contains the IF statement?

2. Study the procedure SPIRAL. Which line contains the recursive call?

Which line contains the IF statement?

The SPIRAL procedure uses two variables, SIDE and ANGLE. Which one is used in the stop rule?

Which procedure input is increased in the recursive call?

3. Diagram the DASHES procedure. Use explicit values for the procedure inputs.

4. What happened when you modified the DASHES procedure?

Data Sheet for Lesson 6

1. Study the procedure SPI.HELPER. Which line contains the recursive command?

What commands are carried out before the recursive call?

What commands are carried out after the recursive call?

2. After typing SQR 1, what is the heading of the turtle?

What is its position relative to its starting position?

After typing SQL 1, what is the heading of the turtle?

What is its position relative to its starting position?

3. Try to predict what the outcome of this procedure will be. Below, draw what you think the turtle will draw.

You may want to diagram the procedure if you are having trouble predicting the outcome.

Name: _____

Data Sheet Masters

4. Compare your predicted outcome with the actual outcome of SQUARES.
5. Predicted outcome of the FANCY.NAME procedure
6. Sketch one copy of the base shape below:

Which picture was created with tail recursion?

Which picture was created using embedded recursion?

Data Sheet for Lesson 7

2. How are NAME and SIDE given their initial values?
3. In the DAYNAME procedure, how does the helper number change?

How does the day name change?

4. Reflections on the DAYNAME procedure
6. Explain global and local variables.

Name: _____

Data Sheet Masters

Data Sheet for Lesson 8

5. Record your reflections on the four fractals.

6. Observations on changes in LEVEL

7. Observations on changes in SIDE

Part 4. Procedure Trees and Listings

Lesson 1. Preliminaries

Procedure Listings

```
TO GREET
CS
GETNAME
DIRECTIONS
END

TO BOX :SIDE
PD
REPEAT 4 [FORWARD :SIDE * 7 RIGHT 90]
PRINT []
(PRINT [I drew a square with side 7 x ] :SIDE "=
      7 * :SIDE)
END

TO GETNAME
SHOW [Please type in your first name]
MAKE "NAME READWORD
END

TO READWORD
OUTPUT FIRST READLISTCC
END

TO DIRECTIONS
REPEAT 8 [PRINT []]
(PRINT "Hi :NAME)
REPEAT 6 [JUMP]
PRINT []
PRINT [Let's draw a square!]
PRINT []
PRINT [Type the command BOX followed by a space
      and a number from 1 to 9]
END
```

```
TO JUMP
PU
FORWARD 10
WAIT 2
BACK 10
WAIT 2
END

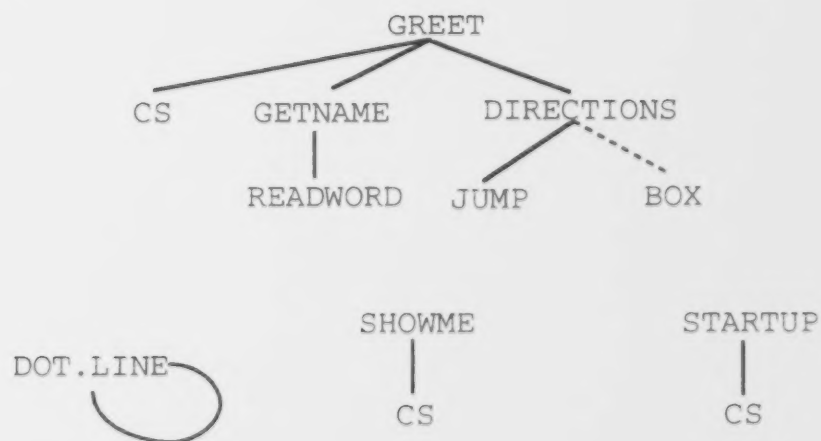
TO DOT.LINE
PD FORWARD 3
PU FORWARD 6
DOT.LINE
END

TO SHOWME
CS
PRINT [TO DOT.LINE]
PRINT [PD FORWARD 3]
PRINT [PU FORWARD 6]
PRINT [DOT.LINE]
PRINT [END]
END

TO STARTUP
CS
END

TO CS
RG
CT
END
```

Procedure Trees



Lesson 2. Embedded Procedures

Procedure Listings

```
TO COUNT.BOX
PRINT [1 2 3 4 5]
PRINT []
WAIT 50
BOX
WAIT 10

PRINT []
PRINT [5 4 3 2 1]
END

TO BOX
RIGHT 90
REPEAT 4 [FORWARD 50 RIGHT 90]
LEFT 90

END

TO CS
CT
RG
END
```

Procedure Tree

```
COUNT.BOX
|
BOX
```

Lesson 3. Simple Recursion

Procedure Listings

```
TO DOT.LINE
  PD FORWARD 3
  PU FORWARD 6
  DOT.LINE
END

TO DUM.DUM
  NOTE1
  NOTE2
  NOTE3
  DUM.DUM
END

TO NOTE1
  (INSERT [dum] CHAR 32)
  TONE 262 5
  END

TO NOTE2
  (INSERT [dum] CHAR 32)
  TONE 294 5
  END

TO NOTE3
  PRINT [hum]
  TONE 332 7
  END

TO CS
  CT
  RG
  END
```

Procedure Trees



Lesson 4. Recursion Involving Procedure Inputs

Procedure Listings

```
TO CS
CT
RG
END
```

```
TO SPIRAL :SIDE :ANGLE
FORWARD :SIDE
RIGHT :ANGLE
SPIRAL :SIDE + 2 :ANGLE
END
```

```
TO SPI.COLOR :SIDE :COLOR :ANGLE
SETC :COLOR
FORWARD :SIDE
RIGHT :ANGLE
SPI.COLOR :SIDE + 2 :COLOR + 1 :ANGLE
END
```

```
TO SPI.HELPER :SIDE :COLOR :ANGLE
IF :SIDE > 60 [STOP]
SETC :COLOR
PRINT []
PRINT []
(PRINT [I'm helper # ] :COLOR)
WAIT 15
FORWARD :SIDE
RIGHT :ANGLE
WAIT 15
CT
```

```
WAIT 10
SPI.HELPER :SIDE + 2 :COLOR + 1 :ANGLE
END
```

Procedure Trees

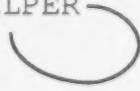
SPIRAL



SPI.COLOR



SPI.HELPER



Lesson 5. Stop Rules

Procedure Listings

```
TO DOT.LINE :COUNT
IF :COUNT > 300 [STOP]
PD FORWARD 3
PU FORWARD 6
DOT.LINE :COUNT + 1
END

TO SPIRAL :SIDE :ANGLE
IF :SIDE > 100 [STOP]
FORWARD :SIDE
RIGHT :ANGLE
SPIRAL :SIDE + 2 :ANGLE
END

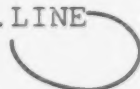
TO DASHES :COLOR
IF :COLOR > 4 [STOP]
SETC :COLOR
PD FORWARD 10
PU FORWARD 15
DASHES :COLOR + 1

END

TO CS
CT
RG
END
```

Procedure Trees

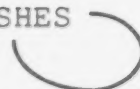
DOT.LINE



SPIRAL



DASHES



Lesson 6. Embedded Recursion

Procedure Listings

```
TO SPI.HELPER :SIDE :COLOR :ANGLE
IF :COLOR > 12 [MOVE.TO.FINISH STOP]
IDENTIFY.AND.DRAW :COLOR
SPI.HELPER :SIDE + 2 :COLOR + 1 :ANGLE
IDENTIFY.AND.FINISH :COLOR
END
```

```
TO IDENTIFY.AND.DRAW :COLOR
SETC :COLOR
(PRINT [I'm helper # ] :COLOR)
WAIT 15
FORWARD :SIDE
RIGHT :ANGLE
WAIT 25
CT
END
```

```
TO IDENTIFY.AND.FINISH :COLOR
SETC :COLOR
(PRINT [I'm helper # ] :COLOR)
PRINT [I'm finished!]
WAIT 35
CT
END
```

```
TO MOVE.TO.FINISH
PU
SETPOS [-60 50]
PD
SETH 0
END
```

```
TO SQUARES :COLOR
IF :COLOR > 4 [STOP]
SQR :COLOR
SQUARES :COLOR + 1
SQL :COLOR
END
```

```
TO SQR :COLOR
SETC :COLOR
FORWARD 12
REPEAT 4 [FORWARD 10 RIGHT 90]
END
```

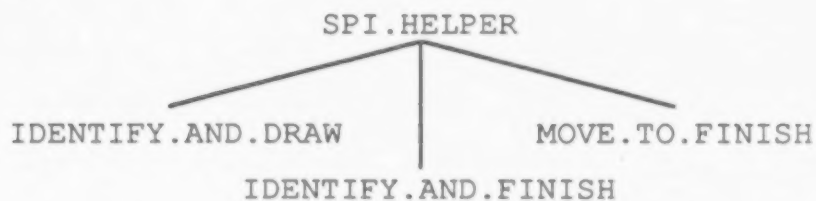
```
TO SQL :COLOR
SETC :COLOR
REPEAT 4 [FORWARD 10 LEFT 90]
```

```
BACK 12
END

TO FANCY.NAME :NAME
IF :NAME = " [STOP]
PRINT :NAME
FANCY.NAME BUTFIRST :NAME
PRINT :NAME
END

TO CS
CT
RG
END
```

Procedure Trees



Lesson 7. Logo Variables

Procedure Listings

```
TO GREET
CS
GETNAME
DIRECTIONS
END

TO GETNAME
SHOW [Please type in your first name]
MAKE "NAME READWORD
END

TO READWORD
OUTPUT FIRST READLISTCC
END

TO BOX :SIDE
PD
REPEAT 4 [FORWARD :SIDE * 7 RIGHT 90]
PRINT []
(PRINT [I drew a square with length 7 x ] :SIDE
  "= 7 * :SIDE)
END

TO DIRECTIONS
REPEAT 8 [PRINT []]
(PRINT "Hi :NAME)
REPEAT 6 [JUMP]
PRINT []
PRINT [Let's draw a square!]
PRINT []
PRINT [Type the command BOX followed by a space
  and a number from 1 to 9]
END

TO JUMP
PU
FORWARD 10
WAIT 2
BACK 10
WAIT 2
END

TO DAYNAME
CS
HT
NAME.DAY 1
END
```

```
TO NAME.DAY :HELPER
GETDAY
IF :DAY = "Saturday [STOP]
(PRINT "helper :HELPER [says it is ] :DAY)
NAME.DAY :HELPER + 1
PRINT "
(PRINT "helper :HELPER [says it is ] :DAY)
WAIT 20
END
```

```
TO GETDAY
SHOW [What day is it?]
MAKE "DAY FIRST READLISTCC
END
```

```
TO MEMORY
HT
MAKE.CONTAINER
HELPERS 1
END
```

```
TO HELPERS :NUMBER
SHOW.MEMORY :NUMBER
IF :NUMBER > 4 [FINISH.UP 5 WAIT 30 STOP]
SHOW.HELPER :NUMBER
DRAW.SQUARE :NUMBER
WAIT 10
HELPERS :NUMBER + 1
FINISH.UP :NUMBER
END
```

```
TO MAKE.CONTAINER
PU
SETPOS [-115 -20]
PD
RIGHT 180
FORWARD 50
LEFT 90
FORWARD 40
LEFT 90
FORWARD 50
PU
SETPOS [-115 -87]
PD
SETC 1
LABEL "COLOR
PU
END
```

```
TO SHOW.HELPER :NUMBER
REPEAT 3 [PRINT []]
```

Part 4. Procedure Trees and Listings

```
REPEAT 5 [INSERT CHAR 32]
(PRINT [Hi! I'm helper # ] :NUMBER)
WAIT 15
PRINT "
(PRINT [I will draw a square with pen color ]
:NUMBER)
WAIT 15
END

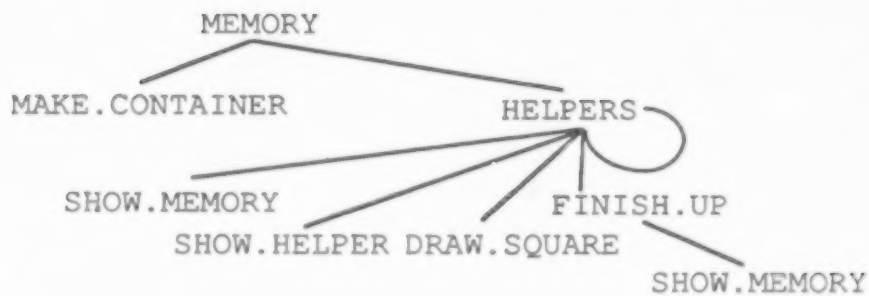
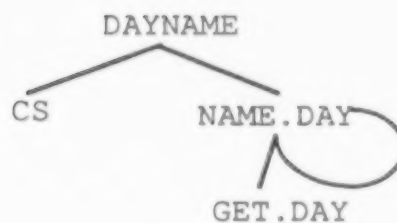
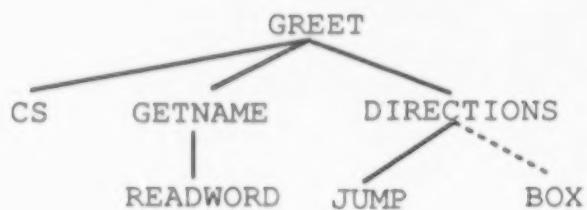
TO SHOW.MEMORY :COLOR
SETX -100
SETY (:COLOR * 9) - 72
PD
SETC 1
LABEL :COLOR
PU
END

TO DRAW.SQUARE :COLOR
PU
SETX 55
SETY -4
SETC :COLOR
PD
REPEAT 4 [FORWARD 20 RIGHT 90]
WAIT 60
CT
PE
REPEAT 4 [FORWARD 20 RIGHT 90]
PU
END

TO FINISH.UP :NUMBER
SETPOS [-60 -62]
PD
LABEL (SENTENCE [helper #] :NUMBER "finishes)
PU
WAIT 30
SHOW.MEMORY :NUMBER
SETPOS [-60 -62]
PD
LABEL (SENTENCE [helper #] :NUMBER "finishes)
PU
WAIT 30
END

TO CS
CT
RG
END
```

Procedure Trees



Lesson 8. Introduction to Fractals

Procedure Listings

```
TO SPIKE :SIDE :LEVEL
IF :LEVEL = 0 [FORWARD :SIDE STOP]
SPIKE :SIDE * .45 :LEVEL - 1
LEFT 90
SPIKE :SIDE * .4 :LEVEL - 1
RIGHT 180
SPIKE :SIDE * .4 :LEVEL - 1
LEFT 90
SPIKE :SIDE * .55 :LEVEL - 1
END

TO POSITION.SPIKE
RG
PU
RIGHT 90
BACK 80
PD
END

TO DRAGONSKIN :SIDE :LEVEL
IF :LEVEL = 0 [FORWARD :SIDE STOP]
RIGHT 45
DRAGONSKIN :SIDE / (2 * SQRT 2) :LEVEL - 1
LEFT 90
DRAGONSKIN :SIDE / SQRT 2 :LEVEL - 1
RIGHT 90
DRAGONSKIN :SIDE / (2 * SQRT 2) :LEVEL - 1
LEFT 45
END

TO SNOWFLAKE :SIDE :LEVEL
IF :LEVEL = 0 [FORWARD :SIDE STOP]
SNOWFLAKE :SIDE / 3 :LEVEL - 1
LEFT 60
SNOWFLAKE :SIDE / 3 :LEVEL - 1
RIGHT 120
SNOWFLAKE :SIDE / 3 :LEVEL - 1
LEFT 60
SNOWFLAKE :side / 3 :LEVEL - 1
END

TO POSITION
RG
PU
BACK 40
PD
END
```

Exploring Recursion With LogoWriter—Teacher's Manual

```
TO SQUAREFLAKE :SIDE :LEVEL
  IF :LEVEL = 0 [FORWARD :SIDE STOP]
  SQUAREFLAKE :SIDE / 3 :LEVEL - 1
  LEFT 90
  SQUAREFLAKE :SIDE / 3 :LEVEL - 1
  RIGHT 90
  SQUAREFLAKE :SIDE / 3 :LEVEL - 1
  RIGHT 90
  SQUAREFLAKE :SIDE / 3 :LEVEL - 1
  LEFT 90
  SQUAREFLAKE :SIDE / 3 :LEVEL - 1
  END

TO POSITION.SQFLK
  RG
  PU
  SETPOS [-135 -80]
  RIGHT 90
  PD
  END

TO CS
  CT
  RG
  END
```

Procedure Trees



Appendix A. *LOGO*WRITER Keys

General-Purpose Keys

	APPLE	IBM	MACINTOSH
Flip Flips a page over	Apple-F	Ctrl-F	⌘-F
Up Moves the cursor up from Command Center	Apple-U	Ctrl-U	⌘-U
Down Moves cursor down into the Command Center	Apple-D	Ctrl-D	⌘-D
Esc Used to leave a page	Esc	Esc	Esc or Clear
Stop Stops program or instruction; returns to Command Center (see also Esc)	Apple-S	Ctrl-Break	⌘-.
Help Gets the Help page	Apple-0	Fn-10	—

Contents Page Keys

	APPLE	IBM	MACINTOSH
Up and Down Arrows Moves cursor up and down through pages that are in the scrapbook	Up & Down Arrows	Up & Down Arrows	Up & Down Arrows
Top of Page Moves the cursor to the top of the Contents list	Apple-Up Arrow	Home	—
Bottom of Page Moves the cursor to the last page name in the Contents list	Apple-Down Arrow	End	—
Return/Enter Chooses the page that the cursor is on	Return	Enter	Return
Erase to End of Line Erases the page named on the line that the cursor is on—permanently!	Apple-6	Fn-6	—

Graphics Keys

	APPLE	IBM	MACINTOSH
Turtle-Move Makes it possible to move the turtle with the Arrow keys; Esc to leave	Apple-9	Fn-9	⌘-9
Label Text typed is added to the picture; use Arrow keys to move; Esc to leave	Apple-8	Fn-8	⌘-8

Appendix B. Suggested Readings

Recursion

- Barclay, Tim; Martin, Kathleen; and Riordon, Tim. (May 1986). "Recursive Tree Designs." *The Computing Teacher*, 13(10), 42-45.
- Billstein, Rick, and Moore, Margaret. (December/January 1983-84). "Recursion, Recursion." *The Computing Teacher*, 11(5), 46-47.
- Clements, Douglas. (December 1989). "Recurrent Recursion Misconceptions." *Logo Exchange*, 8(4), 31-32.
- Harvey, Brian. *Computer Science Logo Style. Volume 1: Intermediate Programming*. Cambridge, MA: MIT Press, 1985.
- Levenick, James R. (October 1990). "Teaching Recursion Before Iteration." *The Computing Teacher*, 18(2), 12-15.
- Logo Computer Systems, Inc. *Apple Logo: Introduction to Programming Through Turtle Graphics*. Canada: LCSi, 1982.
- Lough, Tom. (December/January 1983-84). "A Cure for Recursion." *The Computing Teacher*, 11(5), 34-37.
- Moore, Margaret. (December/January 1983-84). "A Recursion Excursion With a Surprising Discovery." *The Computing Teacher*, 11(5), 49-52.
- Riordon, Tim. "Helping Students With Recursion: Teaching Strategies." (December/January 1983-84). *The Computing Teacher*, 11(5), 38-40.
- Riordon, Tim. "Helping Students With Recursion: Teaching Strategies. Part II: Moving to the Computer." (February 1984). *The Computing Teacher*, 11(7), 59-64.
- Riordon, Tim. "Helping Students With Recursion: Teaching Strategies. Part III: Teaching Students About Embedded Recursion." (March 1984). *The Computing Teacher*, 11(8), 64-69.
- Watt, Molly, and Watt, Daniel. *Teaching With Logo: Building Blocks for Learning*. Menlo Park, CA: Addison-Wesley, 1986.

Yoder, Sharon. *Introduction to Programming in Logo Using LogoWriter*. Eugene, OR: International Society for Technology in Education, 1988.

Fractals

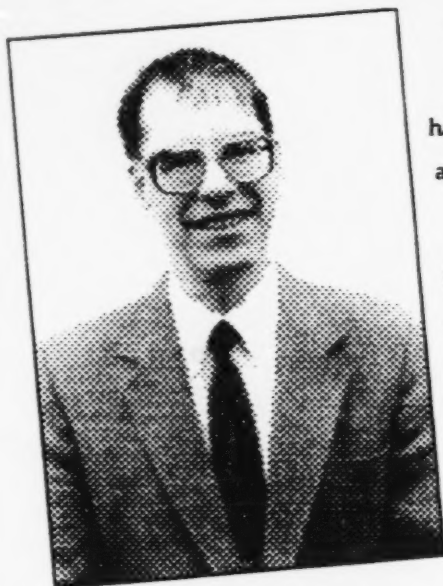
Horney, Mark. "Fractals I: Making Recursion Visible."
(September 1990). *Logo Exchange*, 9(1), 23-29.

Horney, Mark. "Fractals II: Representations." (October 1990).
Logo Exchange, 9(2), 26-29.

Horney, Mark. (November 1990). "Fractals III: Monsters of
Mathematics." *Logo Exchange*, 9(3), 28-31.

Kern, Jane. (March 1991). "Mandelbrot's Sweep of the Koch
Snowflake." *Logo Exchange*, 9(6), 20-24.

Dave Moursund knows...



ISTE Executive Officer Dave Moursund is dedicated to improving education.

Since he first began working with computers in education in 1963, Dave has constantly sought to improve education at all levels through the use of appropriate computer-related technology.

As a continuing part of this effort, Dave's newest book—*The Technology Coordinator*—discusses the goals of education and how teachers and teachers of teachers can join in the enormous change process facing today's schools.

This book was written for educators who want to play a leadership role in the instructional use of technology. Dave uses interviews with technology coordinators to illustrate the needs, roles, and qualifications needed to accomplish the goals of educational technology.

In recent years, Dave has focused much of his writing efforts on problem solving and on staff development. Some of Dave's other books also available from ISTE include *Problem-Solving Models for Computer Literacy*, *LogoWriter for Educators: A Problem Solving Approach*, *Logo PLUS for Educators: A Problem Solving Approach* (co-authored with Sharon Yoder), and *Effective Inservice for Integrating Computer-as-Tool into the Curriculum*.

"Our educational system is not nearly as good as it could be," says Dave. "We know that computer-related technology can make substantial contributions to improving both the content of the curriculum and the instructional delivery system. Such technology can empower students and teachers."

ISTE believes that computer-related technology must become a tool that students and teachers use routinely if students are to be adequately prepared for adult citizenship in our Information Age society.

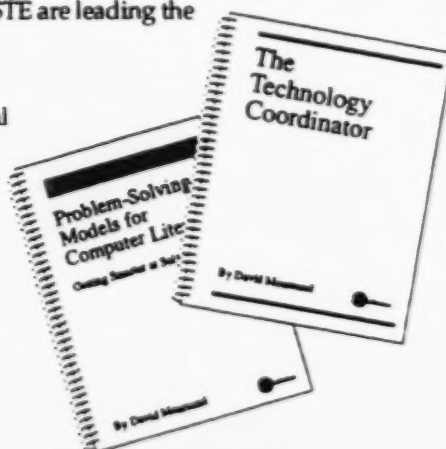
ISTE's wide range of periodicals, publications, projects, and other activities mirrors this vision and commitment.

Call us today to order Dave's latest books, and discover how he and ISTE are leading the way for the educators and school systems of tomorrow.

ISTE. Proudly putting you in touch with the world of educational technology.



International Society for Technology in Education
1787 Agate Street, Eugene, OR 97403-1923
Order Desk: 800/336-5191 Fax: 503/346-5890



ISTE BRINGS THE WORLD OF TECHNOLOGY CLOSER TO YOU.

By drawing from the resources of committed professionals worldwide, ISTE provides support that helps educators like yourself prepare for the future of education.

ISTE members benefit from the wide variety of publications, specialized courseware, and professional organizations available to them.

They also can enjoy exciting conferences, global peer networking, and mind-expanding independent study courses.

So if you're interested in the education of tomorrow, call us today at 800/336-5191.



International Society for Technology in Education
1787 Agate Street, Eugene, OR 97403-1923
Phone: 503/346-4414 Fax: 503/346-5890
Order Desk: 800/336-5191
CompuServe: 70014,2117
Internet: ISTE@Oregon.uoregon.edu
GTE: ISTE.office

AL SOCIETY FOR
TECHNOLOGY

WE'LL PUT YOU IN TOUCH WITH THE WORLD.